CLAIMS

1. A process for translating instructions belonging to a first set of instructions that are pipelined scalar processor instructions into instructions belonging to a second set of instructions that are VLIW processor instructions for execution on a VLIW processor that includes a core, said process comprising the following operations:

providing a first set of registers corresponding to the instructions of said first set;

providing a second set of registers corresponding to the instructions of said second set;

mapping each register of said first set in a corresponding register of said second set designed to emulate the behaviour of the register of said first set, performing a unique independent translation of the instructions of said first set into said second set;

said operations of providing a second set of registers and of mapping being obtained by adding functional units to VLIW processor and keeping said core unaltered; and

performing said translation in the absence of direct access to resources of said core.

2. The process according to claim 1, wherein adding functional units includes adding a translation device external to said core of the VLIW processor,

said translation device intercepting accesses to a storage area reserved to the instructions of the first set.

3. The process according to claim 2, further comprising the operations of forcing a program counter of the VLIW processor to point to a translation memory reserved in the translation device for containing a translation of the instruction belonging to the first instruction set, which in the meantime is decoded.

4. The process according to claim 3, further comprising loading in said reserved translation memory all instructions that constitute the translation of the decoded instruction of the first set.

5. The process according to claim 4, further comprising associating said instructions that constitute the translation of the decoded instruction of the first set a jump-to-link to a next instruction of the first set to be executed, so that all instructions of the first set that are not directly mappable on an instruction of the second set entail a jump to the reserved translation memory and a jump-to-link for loading the next instruction of the first set.

6. The process according to claim 1, further comprising translating all instructions of the first set to which there does not correspond an equivalent single instruction in the second instruction set into an unconditioned GOTO jump.

7. The process according to claim 1, further comprising the operation of forcing a program counter of the VLIW processor to emulate the operation of a program counter of the pipelined scalar processor.

8. The process according to claim 7, wherein said operation of forcing the program counter of the VLIW processor includes forcing said program counter of the VLIW processor to contain a value that is the same as a value of the program counter of a pipelined scalar processor upon loading an instruction belonging to the first instruction set.

9. The process according to claim 8, further comprising executing one of the translated instructions and ending the executing by performing a jump to an address of a next instruction belonging to the first set.

10. The process according to claim 9, wherein emulating the program counter of the pipelined scalar processor includes using a counter register, incrementing said counter register so that each instruction that accesses said counter register during the execution step will have a behaviour that is coherent with the execution on the pipelined scalar processor, and decrementing said counter register for pointing to the next instruction belonging to the first instruction set.

11. The process according to claim 10, wherein said counter register is incremented by a value eight and decremented subsequently by a value four.

12. The process according to claim 11, wherein, for the instructions belonging to the first set that have as destination the program counter, the pointing to the next instruction is obtained by loading into a link register of the VLIW processor the updated value of the counter register and by making an unconditioned GOTO link jump.

13. The process according claim 1 wherein adding functional units includes adding a translation device external to said core of the VLIW processor, said translation device intercepting accesses to a storage area reserved to the instructions of the first set, the process further comprising allowing a program counter of the VLIW processor to evolve freely in the absence of jumps.

14. The process according to claim 13, wherein said translation device is designed to execute the operations of intercepting the accesses to the storage area reserved to the instructions of the first set and to control a set of pointer registers for deciding whether to execute subsequent instructions as instructions of the second set or as instructions of the first set to be emulated.

15. The process according to claim 14, wherein said translation device is inactive until the core of the VLIW processor executes instructions belonging to said

second set and refers the accesses-to-memory to an instruction cache memory, and said translation device is activated when there is an access to the storage area reserved for containing the first set.

16. The process according to claim 15, wherein when said translation device is activated, the translation device loads into a selected one of its internal registers belonging to the set of pointer registers an address which is accessed and carries out reading of the instruction from the corresponding storage area.

17. The process according to claim 16, further comprising the operations of:

translating said instruction read from the reserved storage area and storing said instruction;

allocating an execution window, to which are referred all accesses to memory addresses that start from a current value of the program counter of the VLIW processor and cover an area equal to one occupied by the translation;

reading, using the core of the VLIW processor, the first instruction of the translation from the storage area reserved in the translation device;

incrementing the selected register by a value four, to point to the next instruction of the first instruction set; and

closing said execution window after reading a last instruction of the first set to be translated, and, if at the next access to memory the selected register points outside the reserved storage area, deactivate the translation device.

18. The process according to claim 17, further comprising, in the presence of jumps in a program consisting of the instructions of the first set, rewriting said selected register by a store-word operation contained in the translation of the instructions of the first set into instructions of the second set.

19. The process according to claim 1 wherein the operation of translation on the VLIW processor starts with verification of an execution condition, which includes evaluating one or more flags present in a status register.

20. A translator device for translating instructions belonging to a first instruction set that are pipelined scalar processor instructions into instructions belonging to a second instruction set that are VLIW processor instructions for execution on a VLIW processor that includes a core, said translation device comprising

a translation subsystem designed to receive at input an instruction of the first instruction set and supply at output a translation comprising one or more instructions of the second instruction set;

a translation memory coupled to the translation subsystem and structured to store said translation; and

a control device for taking said translation from said translation memory and supplying it to the core of said VLIW processor.

21. The device according to claim 20, wherein said translation subsystem operates based on a code table stored in said translation memory.

22. The device according to claim 21, wherein the device is connected between the core of said VLIW processor and an instruction-cache memory of said VLIW processor, which operates on a first memory containing instructions of the first set and a second memory containing instructions of the second set, and wherein the control device intercepts accesses of the core of the VLIW processor to said first and second memories.

23. The device according to claim 22, further comprising a set of pointer registers at least in part designed for controlling access to said memories.

24. A computer program product, directly loadable into a memory of a digital computer and comprising software code portions for performing, when the product is run on the computer, a process for translating instructions belonging to a first set of instructions that are pipelined scalar processor instructions into instructions belonging to a second set of instructions that are VLIW processor instructions for execution on a VLIW processor that includes a core, said process comprising the following operations:

providing a first set of registers corresponding to the instructions of said first set;

providing a second set of registers corresponding to the instructions of said second set;

mapping each register of said first set in a corresponding register of said second set designed to emulate the behaviour of the register of said first set, performing a unique independent translation of the instructions of said first set into said second set;

said operations of providing a second set of registers and of mapping being obtained by adding functional units to VLIW processor and keeping said core unaltered; and

performing said translation in the absence of direct access to resources of said core.